CMSC 733 Homework 1
AutoCalib

Nishad Kulkarni
A. James Clark, School of Engineering
University of Maryland, College Park
UID- 117555431, Email: nkulkar2@umd.edu

***Abstract*—This is the submission for the second homework for the subject CMSC 733 for Spring 2022 semester.**

I. Directory Structure and Library versions

Drawn Below is the directory structure for this submission. This will be found after extracting the contents `nkulkar2_hw1.zip`. The assignment codes were written in Python(3.8.10) and used the packages numpy(1.22.3), matplotlib(3.5.1), scipy(1.7.1), and opencv(4.5.5, opencv-contrib-python).

```
nkulkar2_hw1
├── Code
│   └── Wrapper.py
├── Output
│   └── op.txt
├── README.md
└── nkulkar2_hw1_report.pdf
```

II. Initial Intrisic and Extrinsic parameter calculations

First the K Matrix is calculated using the methods detailed in Zhang's paper. For This following pipeline is followed:

1) Read and store images in a list.
2) Get checkerboard corners using `cv2.findChessboardCorners()`.
3) Assume extreme corners for world as [21,21],[193.5,21.5],[21,129], and [193.5,129].
4) Get Homography by using these coordinates in the world as the corresponding coordinates for index 0,8,53, and 45th corners respectively.
5) Find 'B' Matrix using all the Homographies.
6) Find Extrinsic Parameters for each image using calculated 'B' Matrix.
7) Refine Rotation Matrix using Frobenius norm minimization as suggested in Appendix C of the paper.
8) Return combined Extrinsic parameters

The output upto this step looks as follow:

```
B Matrix:
[[ 1.51059843e-07 -3.69160636e-11
-1.16674367e-04] [-3.69160636e-11
1.53592657e-07 -2.08997173e-04]
[-1.16674367e-04 -2.08997173e-04
9.99999971e-01]]

v0= 1360.9094462475925
l= 0.6254189475076539
alpha= 2034.7497469801788
beta= 2017.9030799157126
gamma= 0.49313594636538627
u0= 772.7044068865666

K Matrix before optimization:
[[2.03474975e+03 4.93135946e-01
7.72704407e+02] [0.00000000e+00
2.01790308e+03 1.36090945e+03]
[0.00000000e+00 0.00000000e+00
1.00000000e+00]]
```

We the move to optimizing this matrix using least squares

III. Optimization using Least squares

For Optimizaing the K Matrix we assume initial values as [k1,k2] = 0,0. Moving on, a function to minimize the error for these and previously calculated values was written which is then optimized using `scipy.optimize.least_squares()` function. The resultant K Matrix and k1,k2 values are as follows:

```
Optimized K:
[[2.03387860e+03 4.72538060e-01
7.74910617e+02] [0.00000000e+00
2.01866808e+03 1.36046874e+03]
[0.00000000e+00 0.00000000e+00
1.00000000e+00]]
k1 = -3.1921581538556144e-07
k2 = 5.9777951180610126e-15
```

Finally, these values are then used to undistort the imageset. For undistorting these images, the function `cv2.undistort()` is used. The ouputs are presented in the final section.

IV. Conclusion

As seen in the discussion forum for this semester's Piazza, the gamma values that were obtained by students varied greatly. For having a benchmark to compare, the gold submission from previous year's submission was selected. It was noticed that although the resultant gamma value found in this

asssignment was different, similar value could be attained on wither selecting the first 6 images only or by using RGB converted images, Although, this is might be a fluke, it seemed to be a significant observation worth being mentioned.

V. Reference

CMSC 733, HW1, Student Submissions - Gold submission

VI. Output images

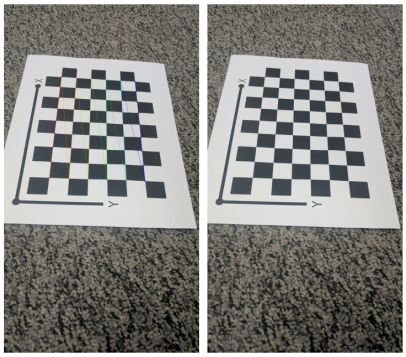

Fig. 1. Cornerized and Undistorted image

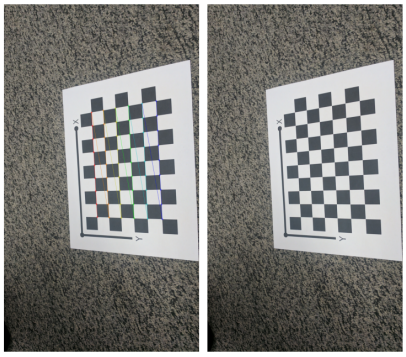

Fig. 2. Cornerized and Undistorted image

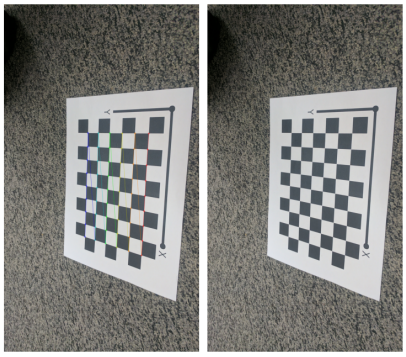

Fig. 3. Cornerized and Undistorted image

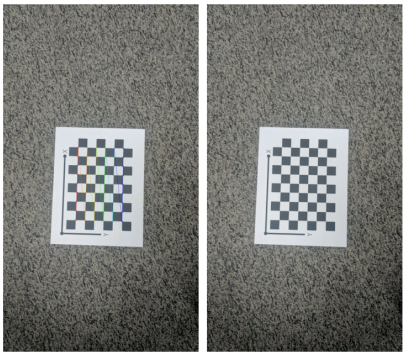

Fig. 4. Cornerized and Undistorted image

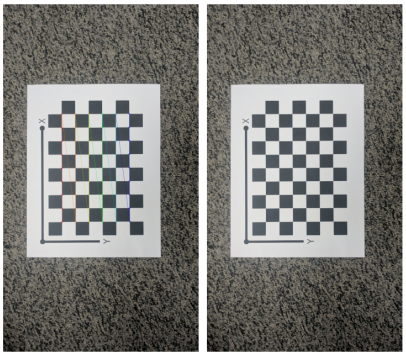

Fig. 5. Cornerized and Undistorted image

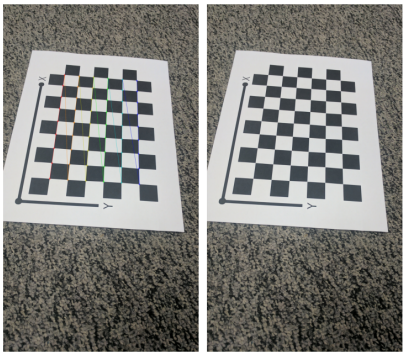

Fig. 6. Cornerized and Undistorted image

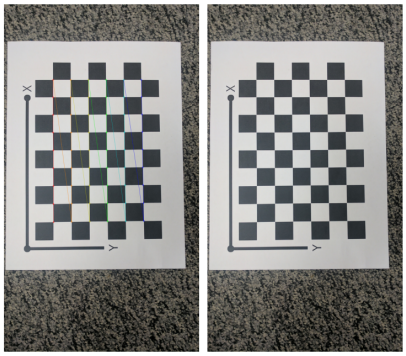

Fig. 7. Cornerized and Undistorted image

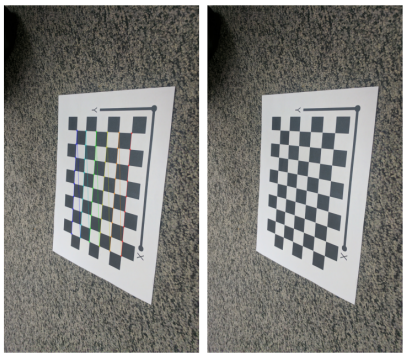

Fig. 8. Cornerized and Undistorted image

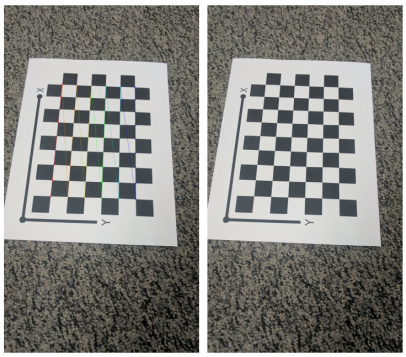

Fig. 9. Cornerized and Undistorted image

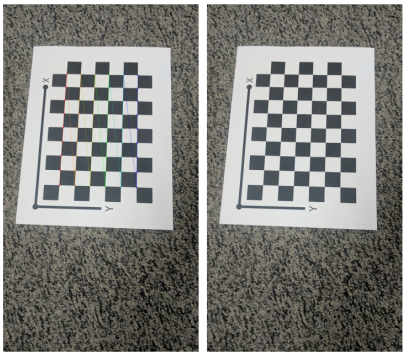

Fig. 10. Cornerized and Undistorted image

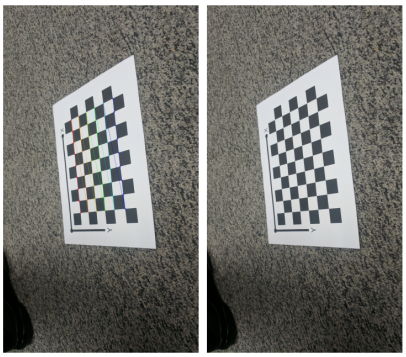

Fig. 11. Cornerized and Undistorted image

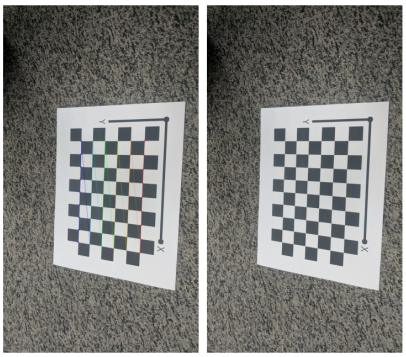

Fig. 12. Cornerized and Undistorted image